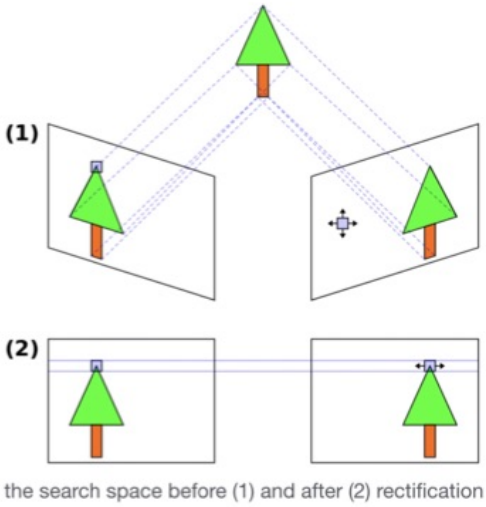
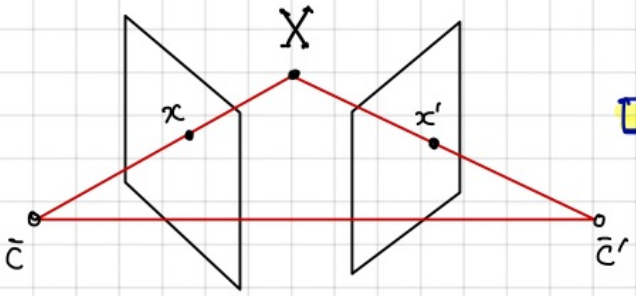
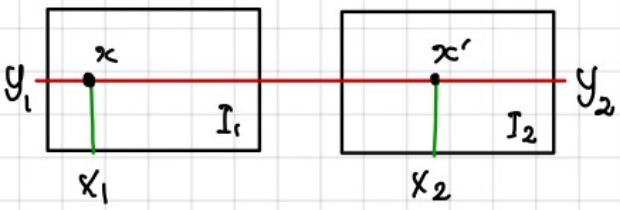
LECTURE 6 : STEREO + MRF

The goal of image rectification:



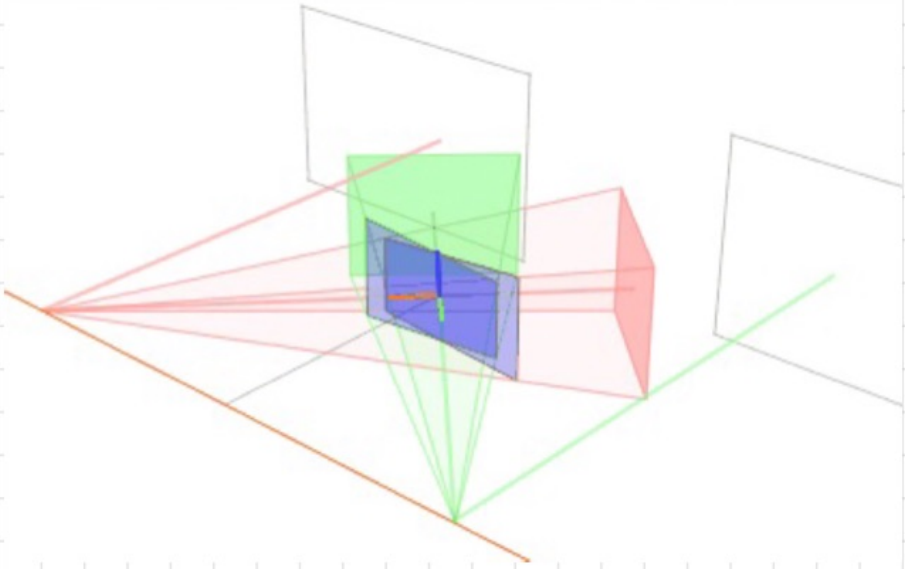
the search space before (1) and after (2) rectification





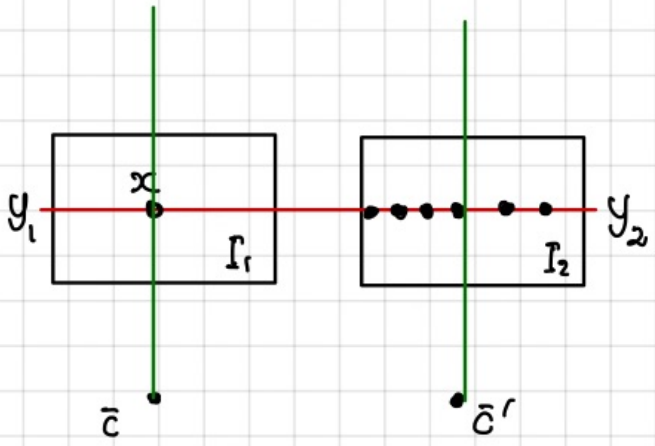
$I_1(x) = I_2(x')$

Image rectification process:



source: Wikipedia

[0] Disparity & Depth

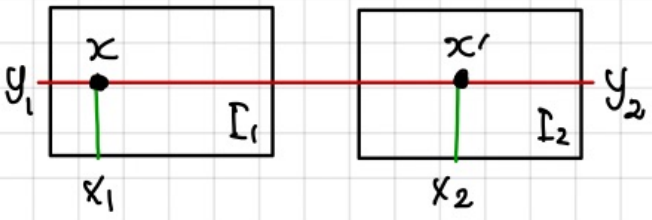


All the points on the red line on the right image are the candidates of a point, x', that make $I_2(x') = I_1(x)$

⇓

Where x & x' represents the same point in the world.

This red line is thus useful to find x' given x. Since, now we don't need to search x' in the entire image of I_2.



$$I_1(x) = I_2(x')$$

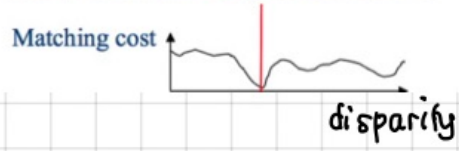
$$I_1(x_1, y_1) = I_2(x_2, y_2)$$

$$y_1 = y_2$$

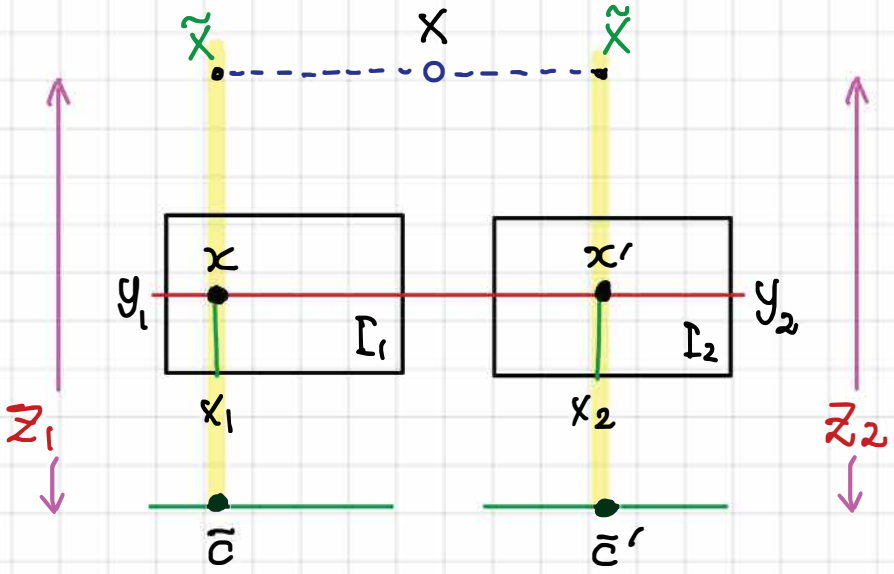
Disparity → $d = |x_1 - x_2|$

The larger the disparity, the smaller the depth : $d \propto 1/Z$

left image

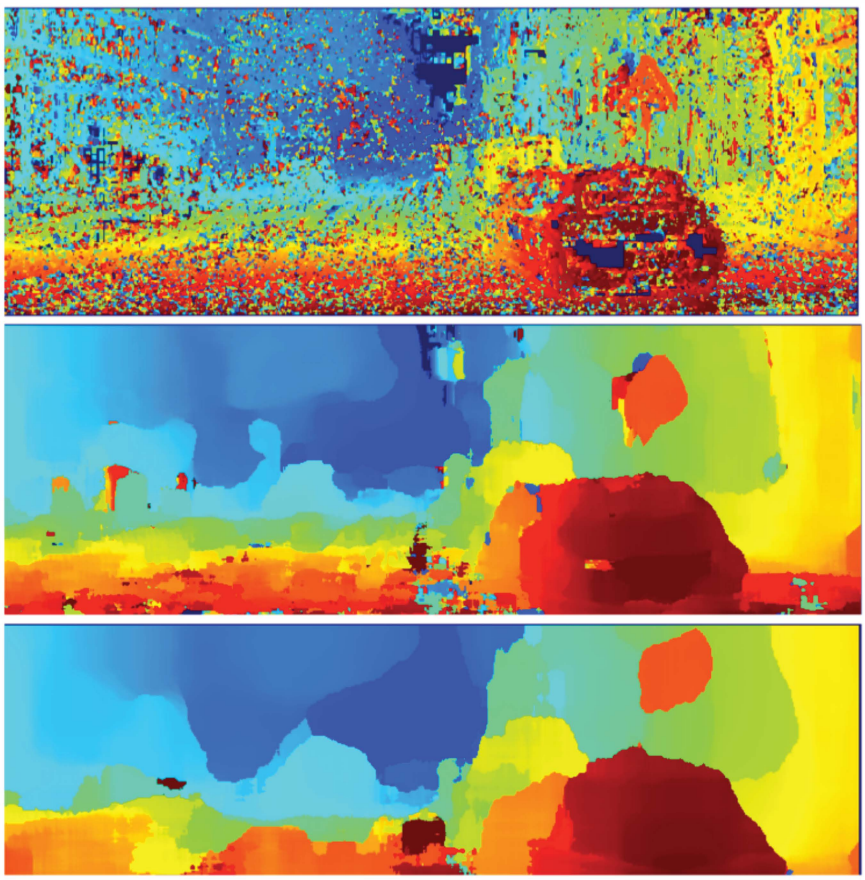


For a pair of rectified images, there is only one disparity or depth map:



[3] Disparity Map from Rectified Images

Pixel or patch based operations:



patch size = 5

patch size = 35

patch size = 85

Problem: small patches → more details but more noise
large patch → less noise but less details

Solution: MRF (Markov Random Fields)

[•] Markov Random Field (MRF)

Markov Random Field (MRF) is one type of graphical models

Motivation : Graphical Models

① An image is a set of numbers representing light intensity / brightness and arranged orderly in a 2D space.

② To make sense those numbers, we need to find the correlations among the numbers.

③ To find the correlations means to formulize the numbers spatially.

④ One of the methods to formulize is through the graphical models

A graphical model is a probabilistic model for which a graph expresses the conditional dependence structure between random variables.

This requires us to understand the basic principles of probabilistic inference.

Basic Probability :

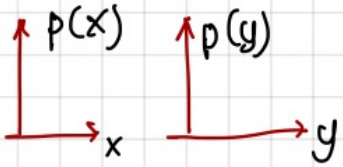
Product rule

Independent variables

$$p(x|y) = p(x)$$
$$p(y|x) = p(y)$$

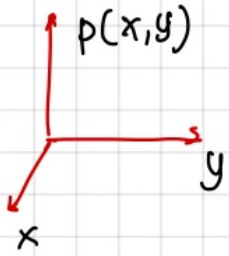
Thus:

$$p(x,y) = p(x)\,p(y)$$



Dependent variables

$$p(x,y) = p(x|y)p(y)$$
$$= p(y|x)\,p(x)$$



Notes:

① $0 \leq p(x) \leq 1$

② $\int p(x)\,dx = 1$

$\sum_x p(x) = 1$

Sum rule (Marginalization)

Two dependent variables x and y :

$$p(x) = \sum_y p(x,y)$$

Example:

x & y are random variables representing the possible faces of a coin

$x, y \in \{\text{Head}, \text{Tail}\}$

Then:

$$p(x=H) = \sum_{y \in \{H,T\}} p(x=H, y)$$
$$= \underbrace{p(x=H, y=H)}_{1/4} + \underbrace{p(x=H, y=T)}_{1/4}$$

Three dependent variables, x, y, z :

$$p(x) = \sum_y \sum_z p(x,y,z)$$

$$p(x,y) = \sum_z p(x,y,z)$$

Bayes' Theorem:

$$p(x,y) = p(x|y)\,p(y) \quad \text{(by product rule)}$$

$$p(x|y) = \frac{p(x,y)}{p(y)}$$

Thus:

$$\boxed{p(x|y) = \frac{p(y|x)\,p(x)}{p(y)}}$$

$$\text{Posterior} = \frac{\text{likelihood} \times \text{prior}}{\text{evidence}}$$

$$\Downarrow$$

$$p(x|d) = \frac{p(d|x)\,p(x)}{p(d)}$$

where:

d = the observed data (the value known to us)

x = the hidden / random variable (the value to estimate)

$p(x|d)$ = the probability of x having a certain value given the data, d.

$p(d|x)$ = the likelihood probability of the data assuming x having a certain value.

$p(x)$ = the prior probability of x

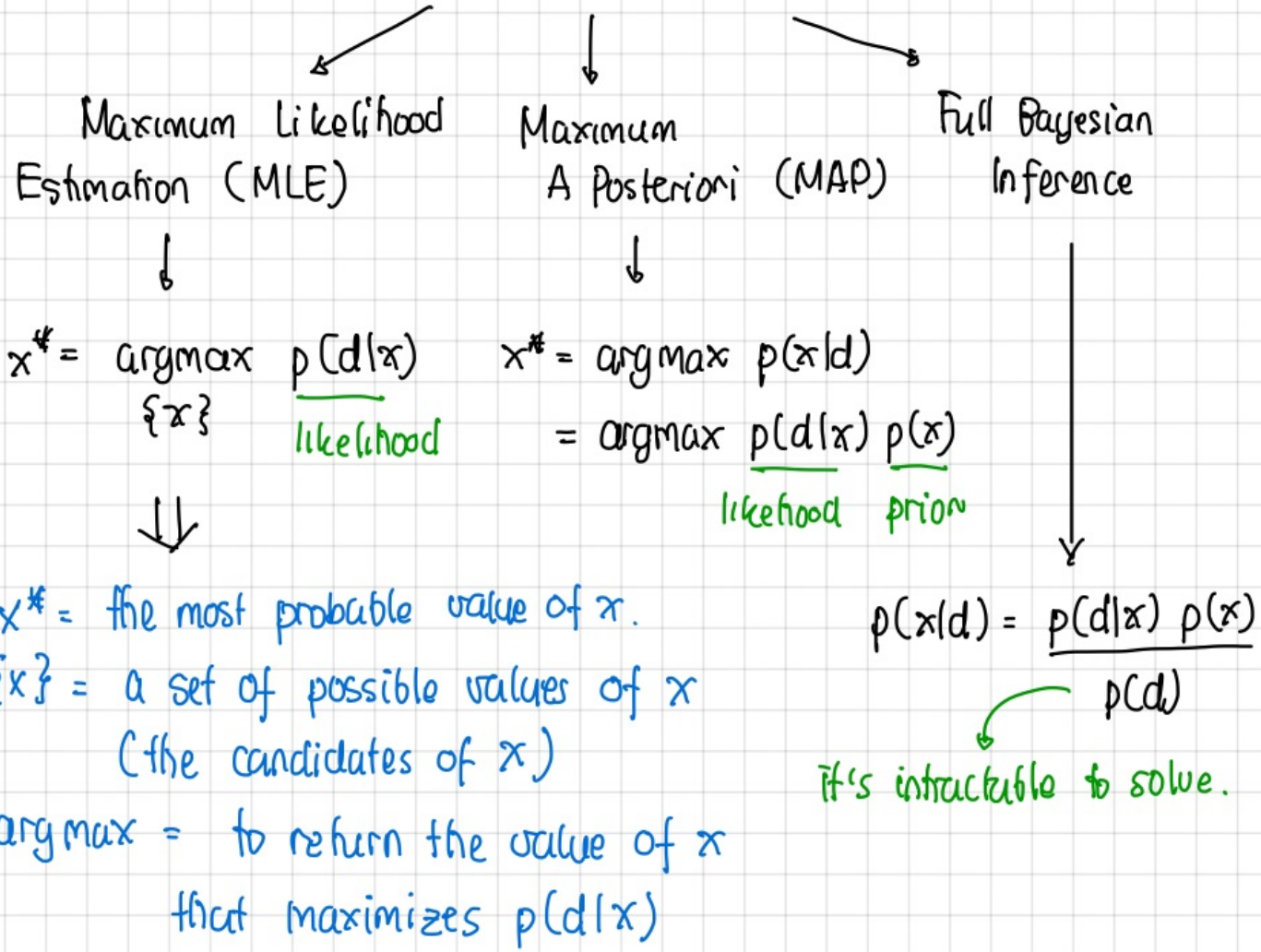
$p(d)$ = the evidence, where:

$$p(d) = \sum_x p(d,x) \quad \text{(by sum rule)}$$

$$= \sum_x p(d|x)\,p(x) \rightarrow \text{expensive to compute if the dimension of } x \text{ is large.}$$

$$p(d) = \int p(d|x)\,p(x)\,dx \rightarrow \text{in most cases it's intractable to integrate.}$$

[•] Probabilistic Inference Methods



Example: According to our observation, a patient has a set of symptomps (= coughing, headache, fever), d.
We also have a set of possible illness: flu, infection and cancer.

The intuitive meaning of MLE:

1. Take each possible illness (e.g. x = flu)
2. Verify if it fits to the symptomps → probability of x.
3. Decide the illness based on the highest probability

[5] Maximum A Posteriori (MAP)

$$x^* = \underset{\{x\}}{\operatorname{argmax}}\ p(x|d) = \underset{\{x\}}{\operatorname{argmax}}\ \frac{p(d|x)\,p(x)}{p(d)}$$

$$x^* = \underset{\{x\}}{\operatorname{argmax}}\ p(d|x)\,p(x)$$

↵ Because $p(d)$ is independent from x.

Notes:

1. The weakness of MLE is that it allows impossible candidates, as it treats candidates uniformly.
2. MAP treats the candidates differently: $p(x=\text{flu}) > p(x=\text{cancer})$
3. Main criticism of MAP is how to determine $p(x)$.

---

[6] Full Bayesian Inference

$$p(x|d) = \frac{p(d|x)\,p(x)}{p(d)}$$

1. This is the ideal and most robust inference that besides providing a prediction on the value of x, it also provides the uncertainty, or how high/low the probability of the prediction.
2. It's possible because of $p(d)$, the evidence. This will normalize the product of $p(d|x)\,p(x)$.
3. However to obtain $p(d)$ is analytically intractable, and in discrete, it can be prohibitively expensive.

[•] Graphical Models: Factorization

Graphical model: A probabilistic model where a graph expresses the conditional dependence structure between random variables.

Conditional Independence:

$$p(x_1 \mid x_2, x_3, x_4) = p(x_1 \mid x_2)$$

↓

It implies that x_1 is independent of x_3 and x_4 given x_2.

↓

Conditional independence is the basic theory of the factorization of joint probability distribution.

Factorization of joint probability

↓

$$p(x_1, x_2, x_3, x_4) = p(x_3 \mid x_2, x_1)\, p(x_2 \mid x_1)\, p(x_1)$$

If x_3 is conditional independent of x_1, then:

$$p(x_1, x_2, x_3) = p(x_3 \mid x_2)\, p(x_2 \mid x_1)\, p(x_1)$$

↓

Factorization reduces the complexity of computing a joint probability

↓

Graphical model is useful due to factorization.

Computing $p(x_1, x_2, x_3)$ means we calculate all possible values of x_1, x_2, x_3. E.g. $x_i \in \{0, 1\}$

then: $x_1 = 0, x_2 = 0, x_3 = 0$
$x_1 = 0, x_2 = 0, x_3 = 1$
$x_1 = 0, x_2 = 1, x_3 = 0$
⋮

} 2^n

If we use factorization:

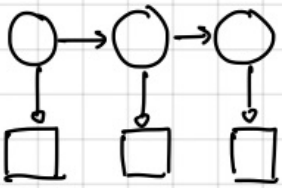
$$p(x_1, x_2, x_3) = p(x_1)\, p(x_2)\, p(x_3)$$

the computation is only: $2n$

[•] Type of Graphical Models

Directed Graphs:

e.g.:



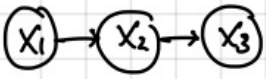
Formulation:

$$p(x_1, \ldots, x_N) = \prod_n p(x_n \mid x_{pa(x_n)})$$

where:

$x_{pa}(x_n)$ = the parents of x_n

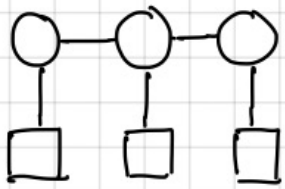
Example:



$$p(x_1, x_2, x_3) = p(x_3 \mid x_2)\, p(x_2 \mid x_1)\, p(x_1)$$

Undirected Graphs

e.g.:



Formulation:

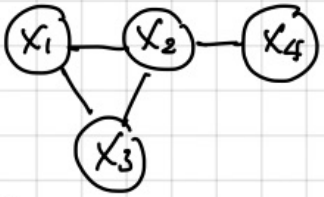
$$p(x_1 \ldots x_N) = \frac{1}{Z} \prod_C \phi_C[x_1 \ldots x_N]$$

where: C = a set of cliques.

Z = a normalization factor.

A clique is a set of mutually connected nodes.

e.g.:



Cliques: $\{\{x_1, x_2, x_3\}, \{x_2, x_4\}\}$

$$\downarrow$$

$$p(x_1, \ldots x_4) = \frac{\phi[x_1, x_2, x_3]\, \phi[x_2, x_4]}{\sum_{x_1}\sum_{x_2}\sum_{x_3}\sum_{x_4} \phi[x_1, x_2, x_3]\, \phi[x_2, x_4]}$$

$$Z = \sum_{x_1}\sum_{x_2}\sum_{x_3}\sum_{x_4} \phi[x_1, x_2, x_3]\, \phi[x_2, x_4]$$

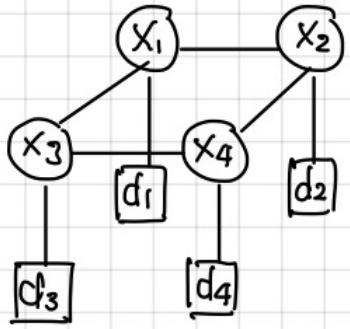
This Z is prohibitively expensive to compute.

Most undirected graphs utilize MAP to ignore Z.

[•] MRF : Formulation

A Markov random field (MRF) is a loopy undirected graph.

e.g. :



Based on the cliques:

$$p(\{x\},\{d\}) = p(x_1 \dots x_4, d_1 \dots d_4)$$

$$= \frac{1}{Z} \phi[x_1,d_1]\,\phi[x_2,d_2]\,\phi[x_3,d_3]\,\phi[x_4,d_4]\,\phi[x_1,x_2]\,\phi[x_2,x_4]\,\phi[x_4,x_3]\,\phi[x_1,x_3]$$

$$= \frac{1}{Z} \prod_{i}^{4} \phi[x_i,d_i] \prod_{j \in N_i} \phi[x_i,x_j]$$

General MRF's formulation:

$$p(\{x\},\{d\}) = \frac{1}{Z} \prod_{i}^{N} \phi[x_i,d_i] \prod_{j \in N_i} \phi[x_i,x_j]$$

Using MAP:

$$\{x\}^{*} = \underset{\{x\}}{\operatorname{argmax}}\ p(\{x\},\{d\}) \propto \underset{\{x\}}{\operatorname{argmax}}\ p(\{x\}|\{d\})$$

$$= \underset{\{x\}}{\operatorname{argmax}} \frac{1}{Z} \prod_{i} \phi[x_i,d_i] \prod_{j \in N_i} \phi[x_i,x_j]$$

$$\propto \underset{\{x\}}{\operatorname{argmax}} \sum_{i} \left[\log \phi[x_i,d_i] + \sum_{j \in N_i} \log \phi[x_i,x_j] \right]$$

$$\{x\}^{*} = \operatorname{argmin} \sum_{i} f_d(x_i,d_i) + \sum_{j \in N_i} f_p(x_i,x_j)$$

[•] MRF : Data and Prior Terms

$$\{x\}^{*} = \arg\min \sum_i \underbrace{f_d(x_i, d_i)}_{\text{Data term}} + \sum_{j \in N_i} \underbrace{f_p(x_i, x_j)}_{\text{Prior term}}$$

Data term

The basic idea of determining the data term:

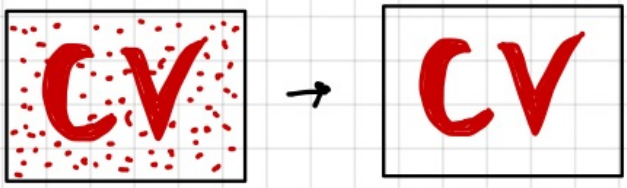
Due to argmin, $f_d(x_i, d_i)$ should produce a smaller value for a more probable x_i's value w.r.t. d_i.

Prior term:

The basic idea of determining the prior term (smoothness prior):

$f_p(x_i, x_j)$ should generate a smaller value when x_i & x_j have the same value.

Example: Denoising



1. $x_i \in \{0, 1\}$ or $x_i = \{B, F\}$ where x_i d_i : $0 \rightarrow$ white, x_i d_i : $1 \rightarrow$ red
2. Prior term: $f(x_i, x_j) = |x_i - x_j|$: We encouraged them to be the same.
3. Data term: The idea is that if d_i is white, x_i should be 0 or B. and if d_i is red, x_i should be 1 or F.

(1) Transform x_i's value to d_i's domain → Because $x_i = \{0, 1\}$ and $d_i = \{0, \ldots 255\}$

$$T(x_i) = \begin{cases} \text{white} = (255, 255, 255) & \text{if } x_i = 0 \\ \text{red} = (255, 0, 0) & \text{if } x_i = 1 \end{cases}$$

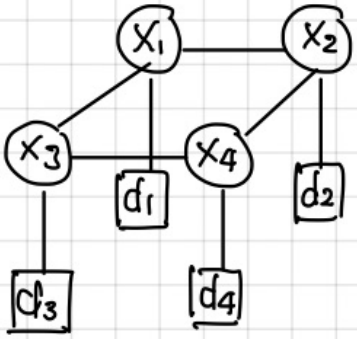
(2) Define a distance function between $T(x_i)$ and d_i.

e.g.: $f_d(x_i, d_i) = f_d(T(x_i), d_i) = |T(x_i) - d_i|$

Since $x_i = \{0, 1\}$, thus there are 2 possible values of f_d:

$f_d(T(x_i = 0), d_i)$ and $f_d(T(x_i = 1), d_i)$.

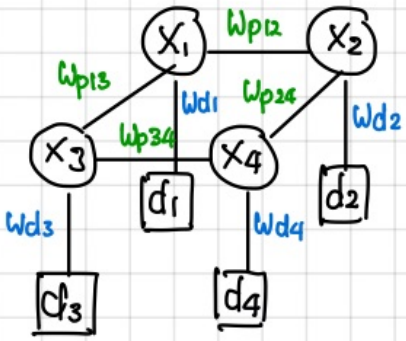
[•] MRF : Graph Optimization



$$\{x\}^{*} = \arg\min \sum_{i} \underbrace{f_d(x_i, d_i)}_{\text{Data term}} + \sum_{j \in N_i} \underbrace{f_p(x_i, x_j)}_{\text{Prior term}}$$

⇓

Having the values of f_d and f_p for every x_i means that we have a complete weighted graph.



⇓

Q: How from this weighted graph, can we estimate the optimum values of $\{x\}$?

A: Graph optimization methods

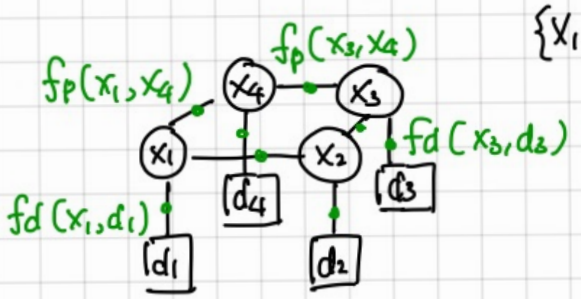
↙ Belief Propagation (BP)

↘ Graphcuts

[•] MRF : Optimization

[i] Motivation

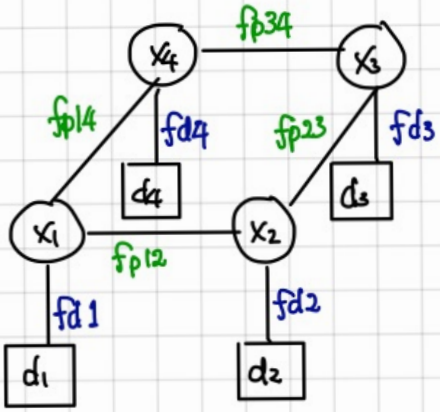
MRF :



$$\{x_1,\dots,x_4\}^* = \underset{\{x\}}{\operatorname{argmin}} \sum_i f_d(x_i,d_i) + \sum_{j\in N_i} f_p(x_i,x_j)$$

the value of each term indicates the values of the edges in the graph.

Having defined the data term (f_d) & the prior term (f_p) for every node (pixel), we can now have a complete graph :



Q : How can we optimize the graph so that we can predict the labels of x_1, x_2, x_3, x_4 ?

Q : What does it mean by 'optimizing' a graph ?

A : Example : x_1 — x_2 & $x_i \in \{0,1\}$

$$\{x_1,x_2\}^* = \underset{\{x_1,x_2\}}{\operatorname{argmin}} f_p(x_1,x_2)$$

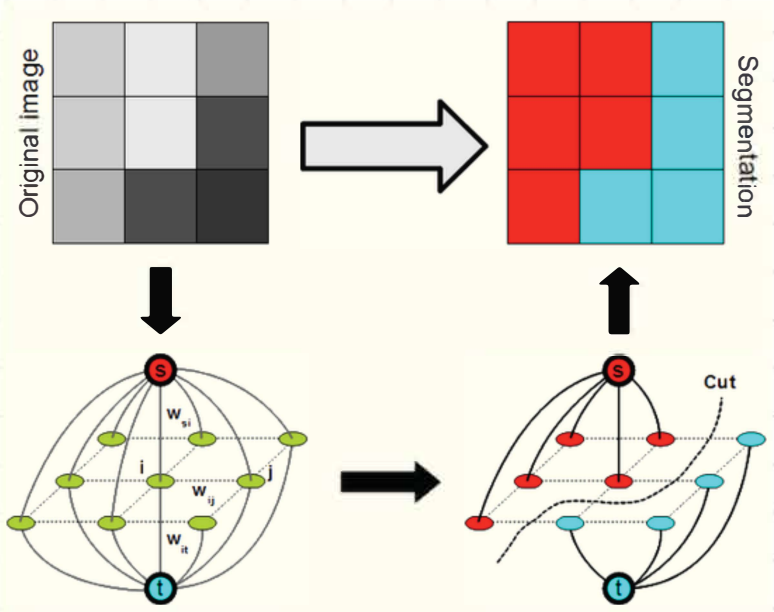
$$= \underset{\{x_1,x_2\}}{\operatorname{argmin}} \left[f_p(x_1=0,x_2=0),\ f_p(x_1=1,x_2=0),\ f_p(x_1=1,x_2=0),\ f_p(x_1=1,x_2=1) \right]$$

We find the values of x_1 & x_2 that minimize $f_p()$, or to choose one of four possible configurations.

[2] Graphcuts

There are two methods of graph optimization: Graphcuts & Belief Propagation. Here, we focus on graphcuts, particularly on binary graphcuts.

Basic concept: Imagine we have a 3x3 image. Some of the pixels belong to the foreground & some belong to the background. However, due to noise/outlier and color variations, we are not sure & thus want to determine which pixels belongs to which label.



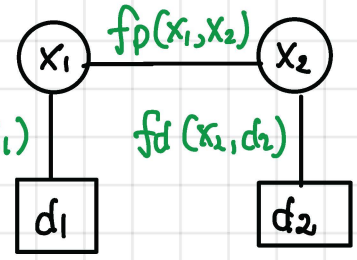
[3] Graphcuts: Data & Prior Terms

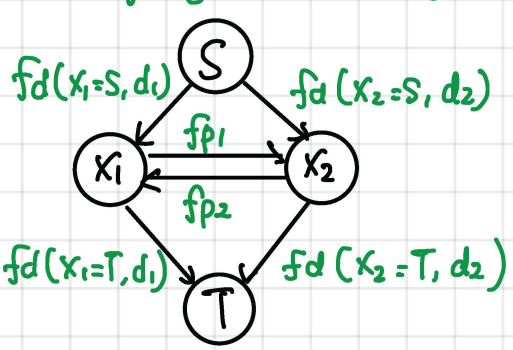
Q: How to assign the data & prior terms in a graph?

A: Suppose we have: x_1 & x_2 where $x_i = \{S, T\}$ (S: foreground, T: background)

and d_1, d_2

MRF:



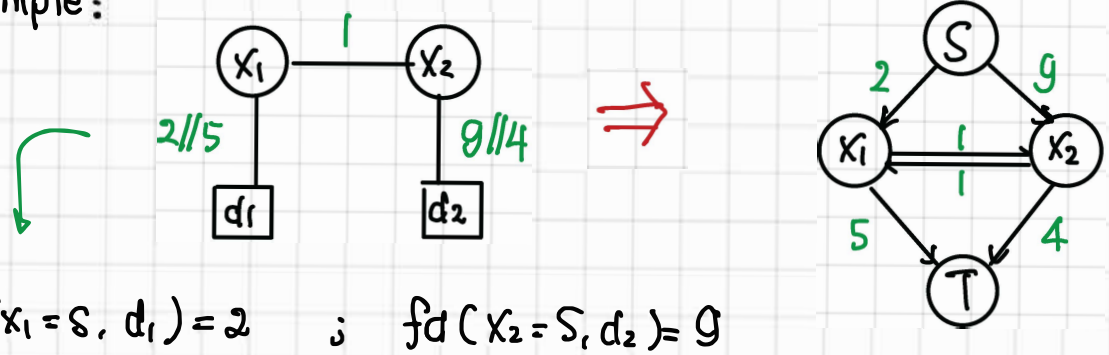


Assume:

$$f_p(x_1 = T, x_2 = S) = f_p(x_1 = S, x_2 = T) \rightarrow f_{p1} = f_{p2}$$

$$f_p(x_1 = T, x_2 = T) = f_p(x_1 = S, x_2 = S) = 0$$

Example:



$fd(x_1 = S, d_1) = 2$; $fd(x_2 = S, d_2) = 9$

$fd(x_1 = T, d_1) = 5$; $fd(x_2 = T, d_2) = 4$

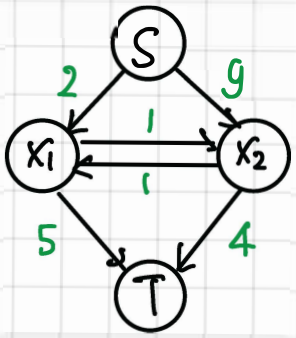
$fp(x_1 = S, x_2 = T) = fp(x_1 = T, x_2 = S) = 1$

$fp(x_1 = S, x_2 = S) = fp(x_1 = T, x_2 = T) = 0$

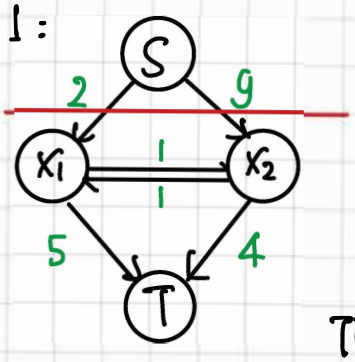
[4] Graphcuts: Max-Flow / Min-cut

Q: How to optimize the graph, so that we know the labels of x_1 and x_2?

A:



cut 1:



Cut cost:

$9 + 2 = 11$

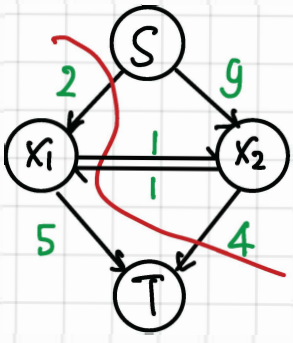
This cut implies:

$x_1 = S$

$x_2 = S$

This is for minimization. For maximization: $x_1 = T$, $x_2 = T$

cut 2:



Cut cost:

$2 + 1 + 4 = 7$

Implying:

$x_1 = S$

$x_2 = T$

Q: Which cut is better?

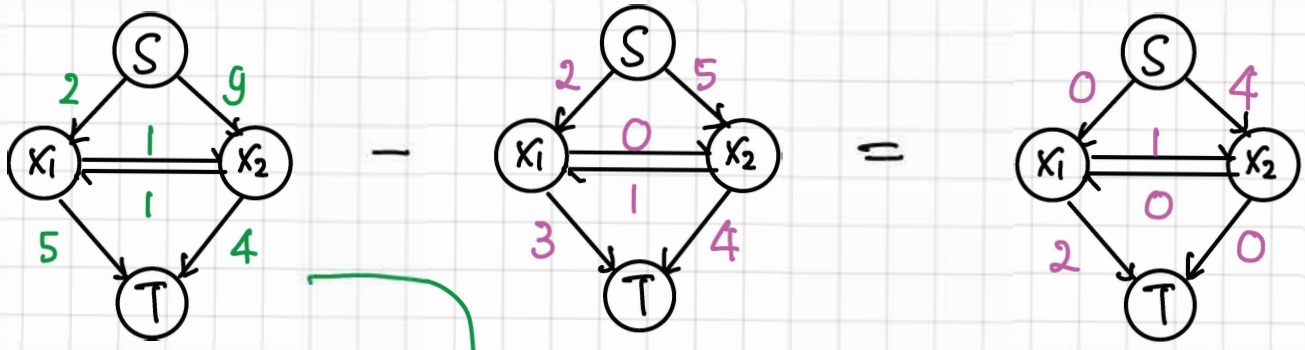
A: Cut 2 is cheaper than cut 1!

Q: How to find the minimum cut for all possible cuts?

[5] Graphcuts: Augmenting Path (the Ford-Fulkerson method)

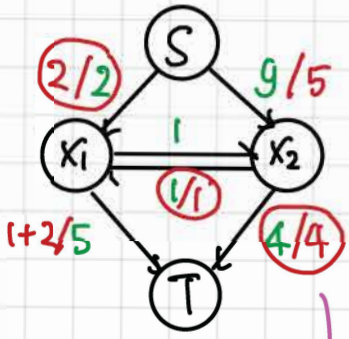
Basic components

(1) Capacity graph — (2) Flow graph — (3) Residual graph

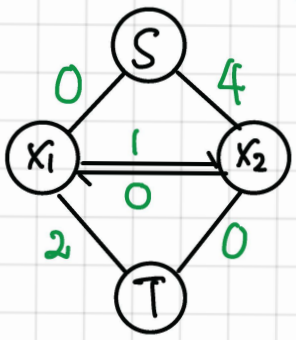


Calculation:

Start from 2:



Residual:

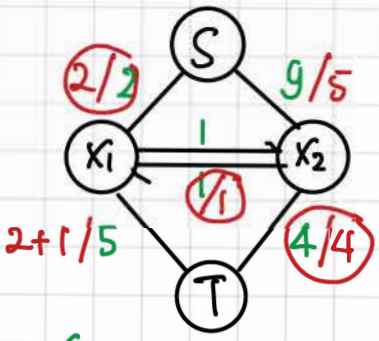


Cut cost = 2+1+4 = 7
(Min-cut)

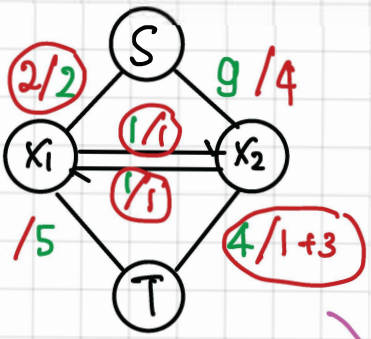
Max-flow = 2+5 = 3+4 = 7

Must be the same!

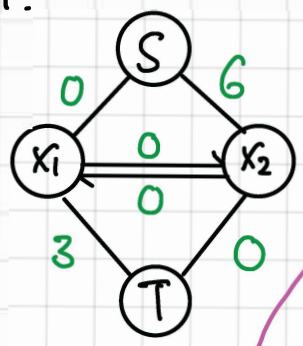
Alternative 1 (Start from 9):



Alternative 2 (from 2 then 1):



Residual:

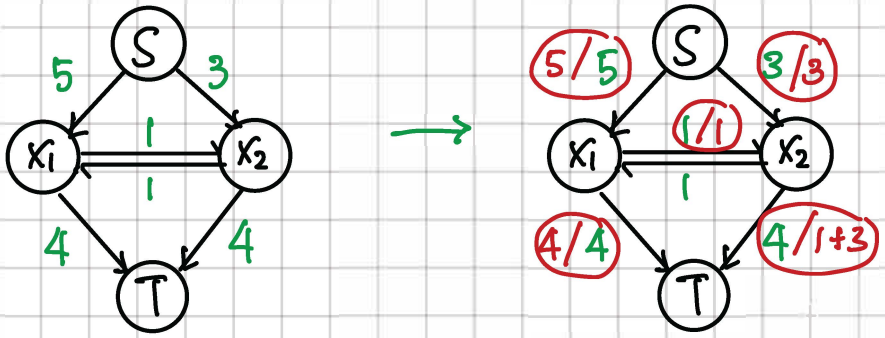


Cut cost = 2+1+1+4 = 8
(Min-cut)

Max-flow = 2+4 = 1+1+1+3 = 6

Other examples:

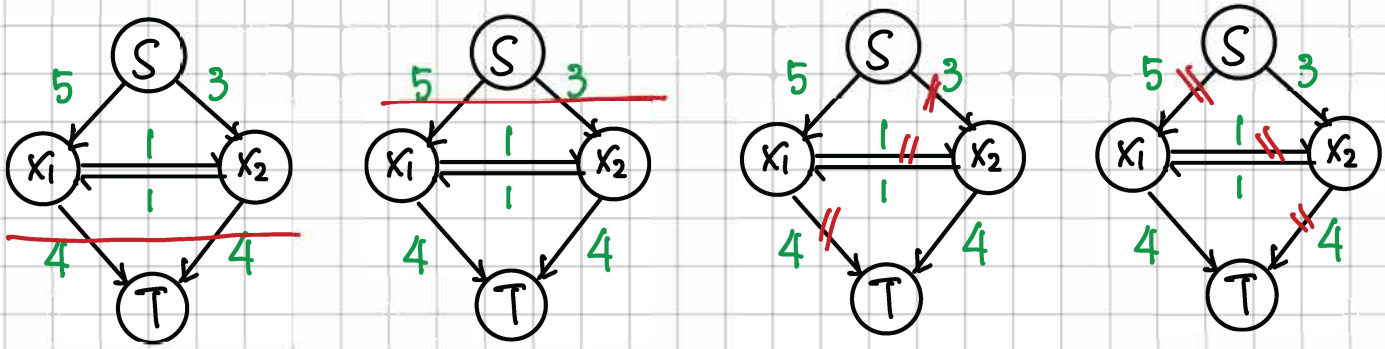
1.



Min cut = 8

Max flow = 8

For this case, there are a few different ways of cutting:



cut cost = 8
$x_1 = x_2 = T$

cut cost = 8
$x_1 = x_2 = S$

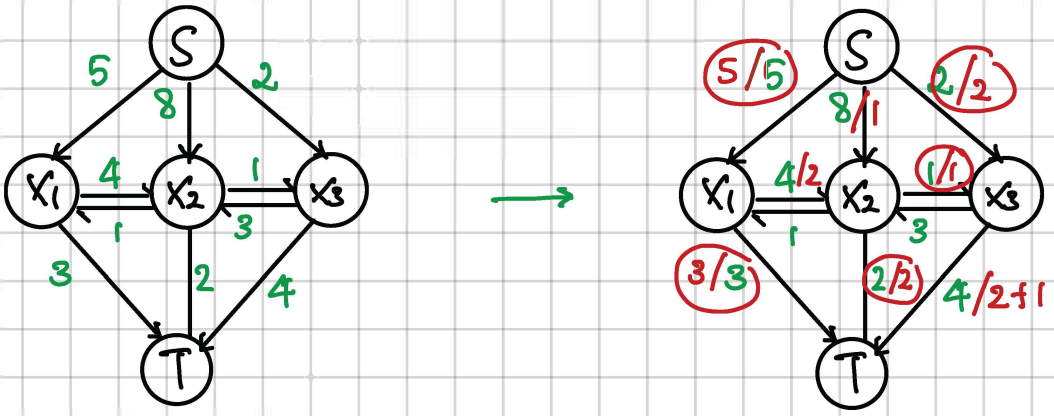
cut cost = 8
$x_1 = T$; $x_2 = S$

cut cost : 10
$x_1 = S, x_2 = T$
(not preffered because the cost is high)

all these labelings are correct.

This implies there are ambiguities in the labeling.

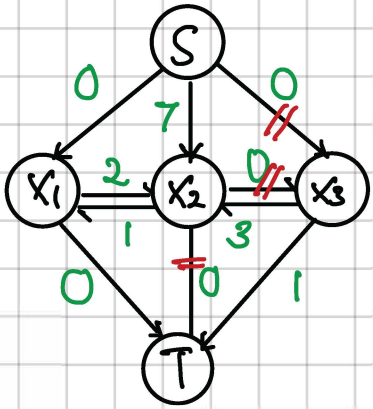
2.



$x_2 = T$, $x_3 = S$ → total cost : 2+1+2 = 5

$x_1 = \begin{cases} S \rightarrow \text{additional cost} = 5 \\ T \rightarrow \text{additional cost} = 3 \end{cases}$

We know the Max-flow = 8, thus $x_1 = T$



[6] Belief Propagation (BP)

Aside from graphcuts, BP is another popular method for graph optimization. It's more general than graphcuts (can accept many labels) but much slower.

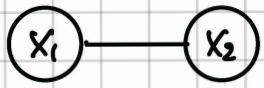
Formulation:

Belief: $b_i(x_i) = k\, \phi_i(x_i) \prod_{j \in N_i} m_{ji}(x_i)$

- k: scalar
- $\phi_i(x_i)$: Data term
- $m_{ji}(x_i)$: to prevent a very small / large value due to the product of m_{ji}

Message passing: $m_{ij}(x_j) = \sum_{x_i} \phi_i(x_i)\, \psi_{ij}(x_i, x_j) \prod_{k \in N_i/j} m_{ki}(x_i)$

- $\psi_{ij}(x_i, x_j)$: Prior term
- N_i/j: Neighbors of i except j.

Example: 

; $x_i = \{F, B\}$

$$b(x_1 = F) = k\, f_d(x_1 = F, d)\, m_{21}(x_1 = F)$$

$$m_{21}(x_1 = F) = \sum_{x_2 \in \{F,B\}} f_d(x_2, d_2)\, f_p(x_2, x_1 = F)$$

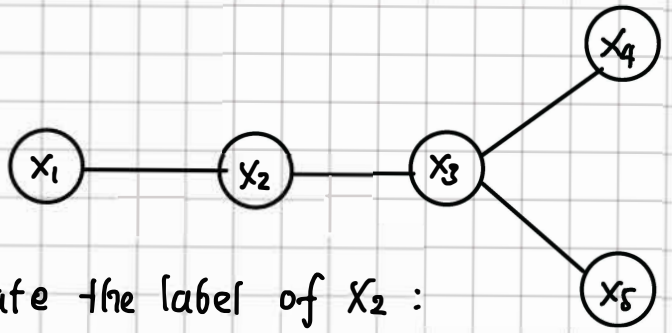
$$= f_d(x_2 = F, d_2)\, f_p(x_2 = F, x_1 = F) + f_d(x_2 = B, d_2)\, f_p(x_2 = B, x_1 = F)$$

Aside from $b(x_1 = F)$, we also compute $b(x_1 = B)$, and compare $b(x_1 = F)$ and $b(x_1 = B)$.

If $b(x_1 = F) > b(x_1 = B)$, then we label $x_1 = B$

For minimization

Another example:



Suppose we want to calculate the label of x_2:

$$b_2(x_2) = k\,\phi_2(x_2)\; m_{12}(x_2)\; m_{32}(x_2)$$

$$m_{12}(x_2) = \sum_{x_1} \phi_1(x_1)\; \psi_{12}(x_1, x_2)$$

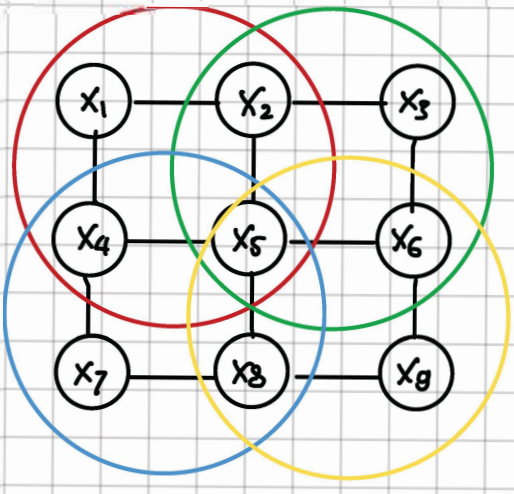
$$m_{32}(x_2) = \sum_{x_3} \phi_3(x_3)\; \psi_{32}(x_3, x_2)\; m_{43}(x_3)\; m_{53}(x_3)$$

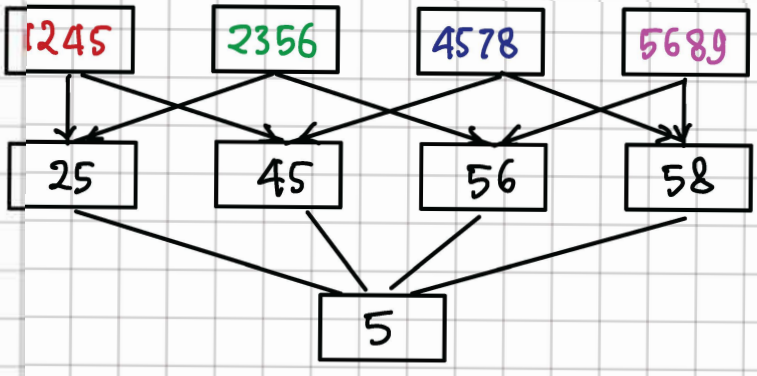
$$m_{43}(x_3) = \sum_{x_4} \phi_4(x_4)\; \psi_{43}(x_4, x_3)$$

$$m_{53}(x_3) = \sum_{x_5} \phi_5(x_5)\; \psi_{53}(x_5, x_3)$$

Note:

» The standard BP works only for a non-loopy graph.
For loopy graphs, one of the solutions is the Generalized BP.





$$b_5(x_5) = k\,\phi_5(x_5)\; m_{2\to5}\; m_{4\to5}\; m_{6\to5}\; m_{8\to5}$$

$$m_{4\to5}(x_5) = k \sum_{x_4} \phi_4(x_4)\; \psi_{45}(x_4, x_5)\; m_{12\to45}\; m_{78\to45}$$